Pattern Recognition

(EE5907)

Song Bai

Email: songbai.site@gmail.com

Outlines

- Unsupervised Feature Extraction (PCA, NMF,…)
- Supervised Feature Extraction (LDA, GE, …)
- Clustering and Applications
- Gaussian Mixture Model and Boosting
- **Support Vector Machine**
- Deep Learning

Some slides are from Xiaojin Zhu

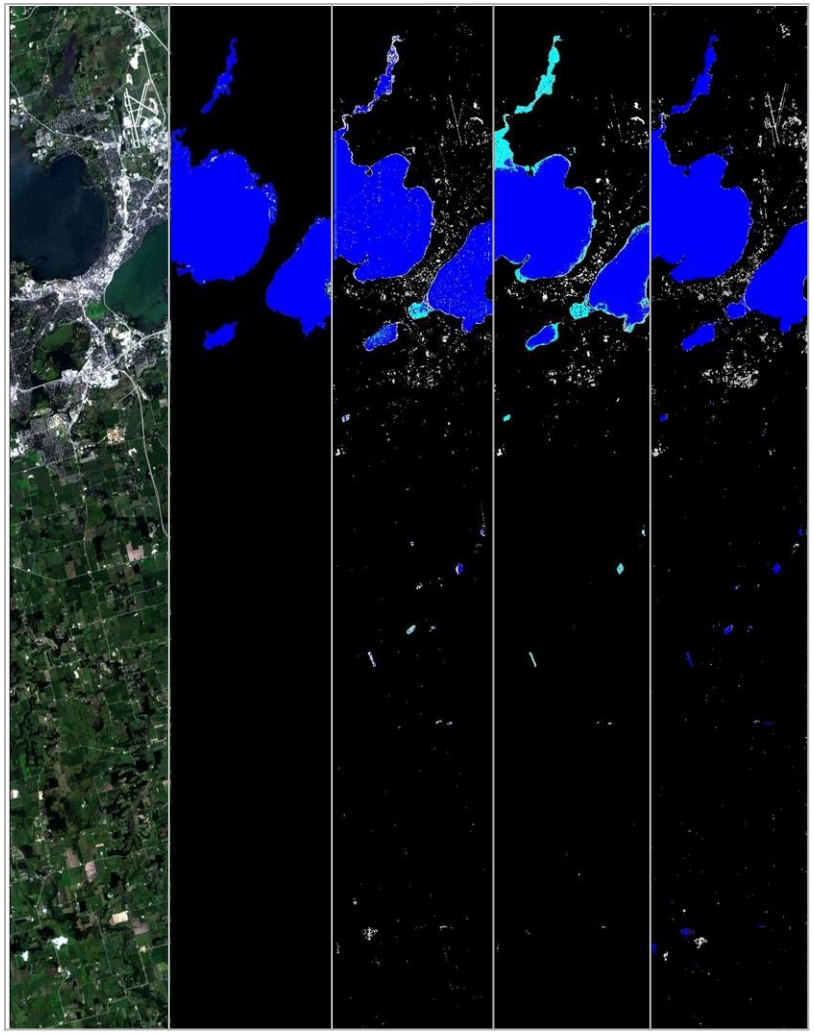
- 识别场景中的土地覆盖区域（土地、冰、水、雪）。
- 三种算法。
§ 科学家手动得出
§ 自动最佳比例
§ 支持向量机（SVM）

Lake Mendota, Madison, WI

- Identify areas of land cover (land, ice, water, snow) in a scene
- Three algorithms:
  - Scientist manually derived
  - Automatic best ratio
  - Support Vector Machine (SVM)

Classifier	Expert Derived	Automated Ratio	SVM
cloud	45.7%	43.7%	58.5%
ice	60.1%	34.3%	80.4%
land	93.6%	94.7%	94.0%
snow	63.5%	90.4%	71.6%
water	84.2%	74.3%	89.1%
unclassified	45.7%		

Lake Mendota, Wisconsin



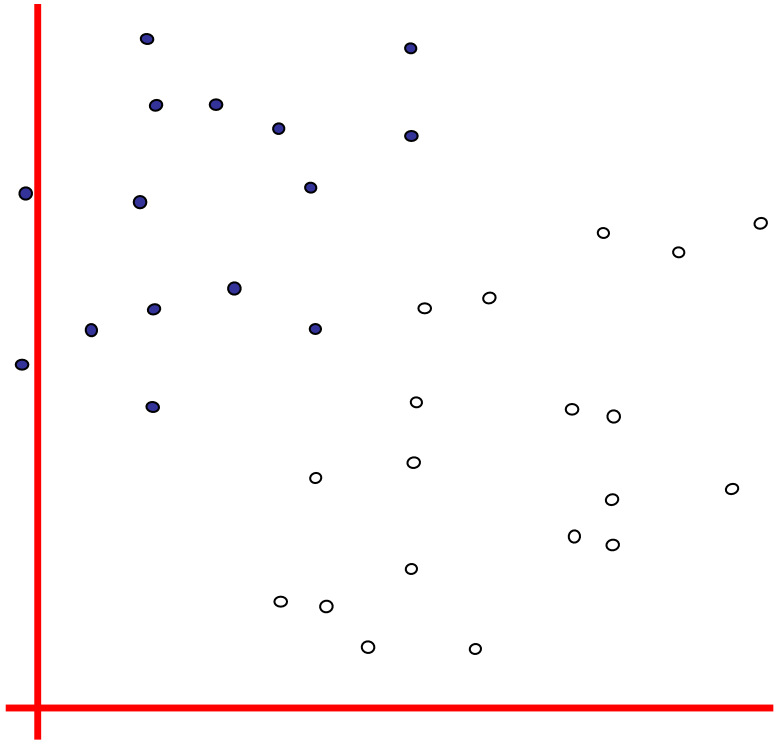
Visible Image | Expert Labeled | Expert Derived | Automated Ratio | SVM

Support vector machines

- The state-of-the-art classifier

Class labels

- denotes +1
- denotes -1

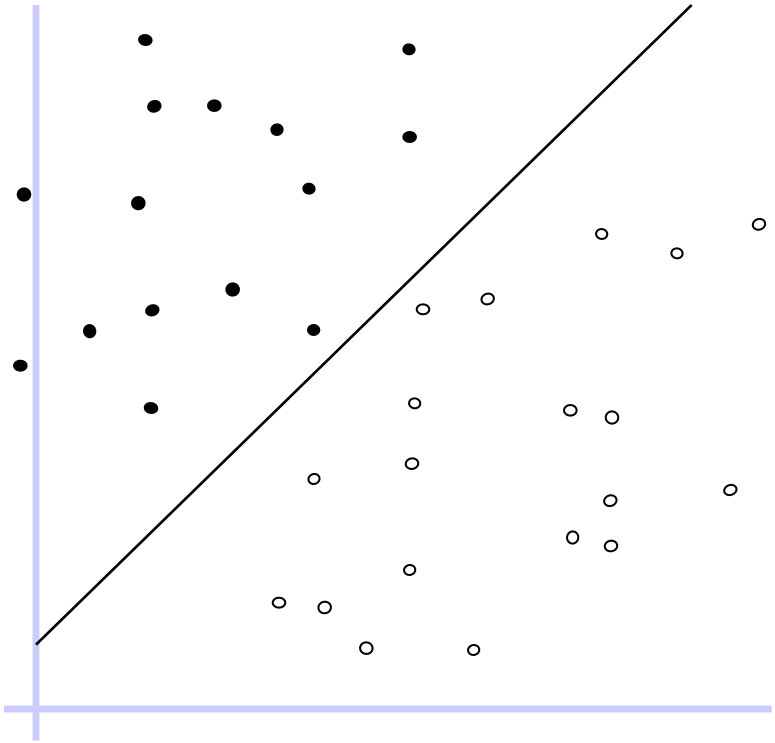


How would you classify this data?

Linear classifier

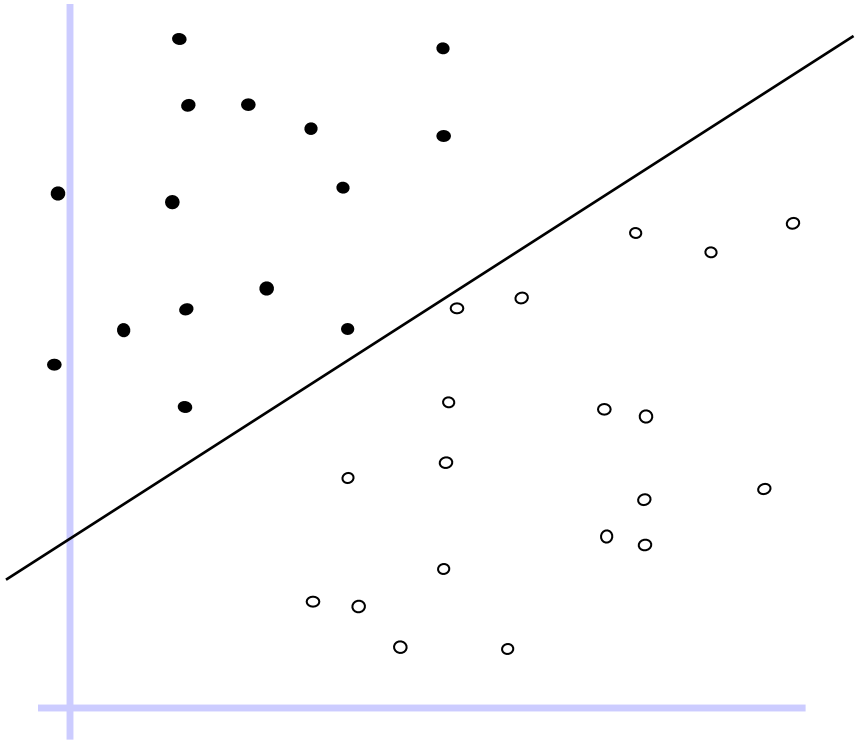
- If all you can do is to draw a straight line, the ‘linear decision boundary’…

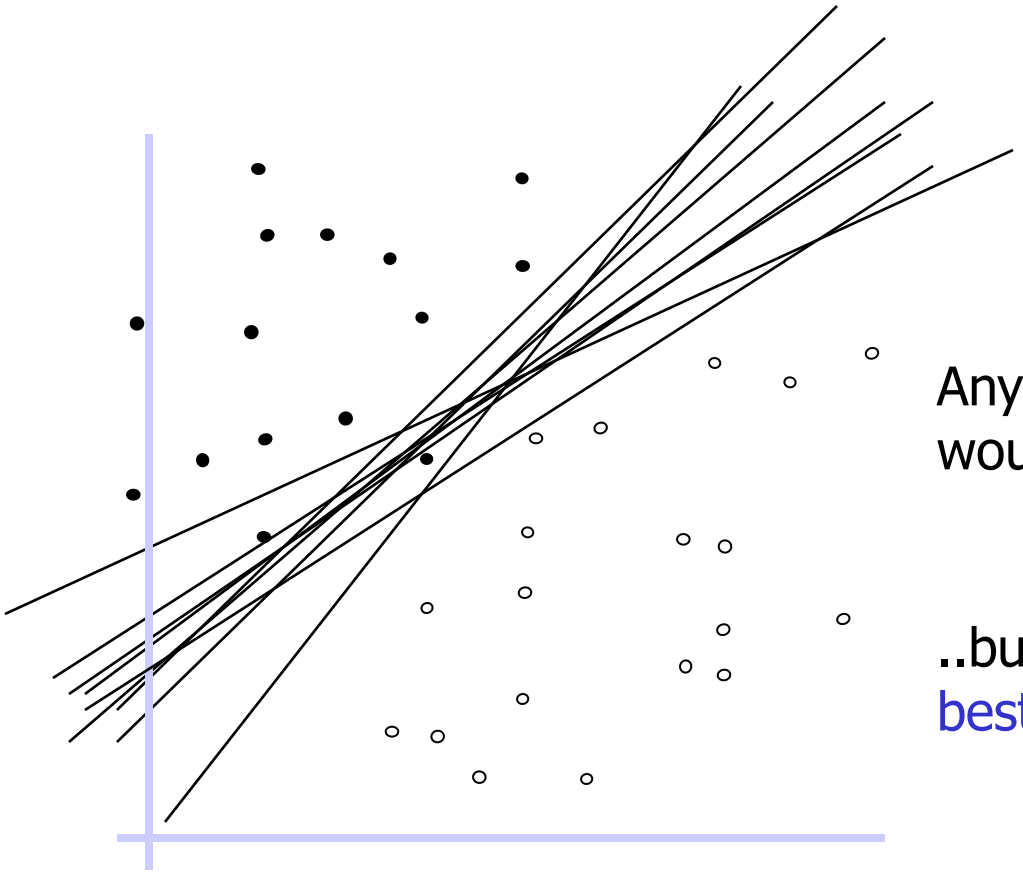
如果你能做的只是画一条直线，"线性决策边界"...



Linear classifier

- Another OK ‘decision boundary’



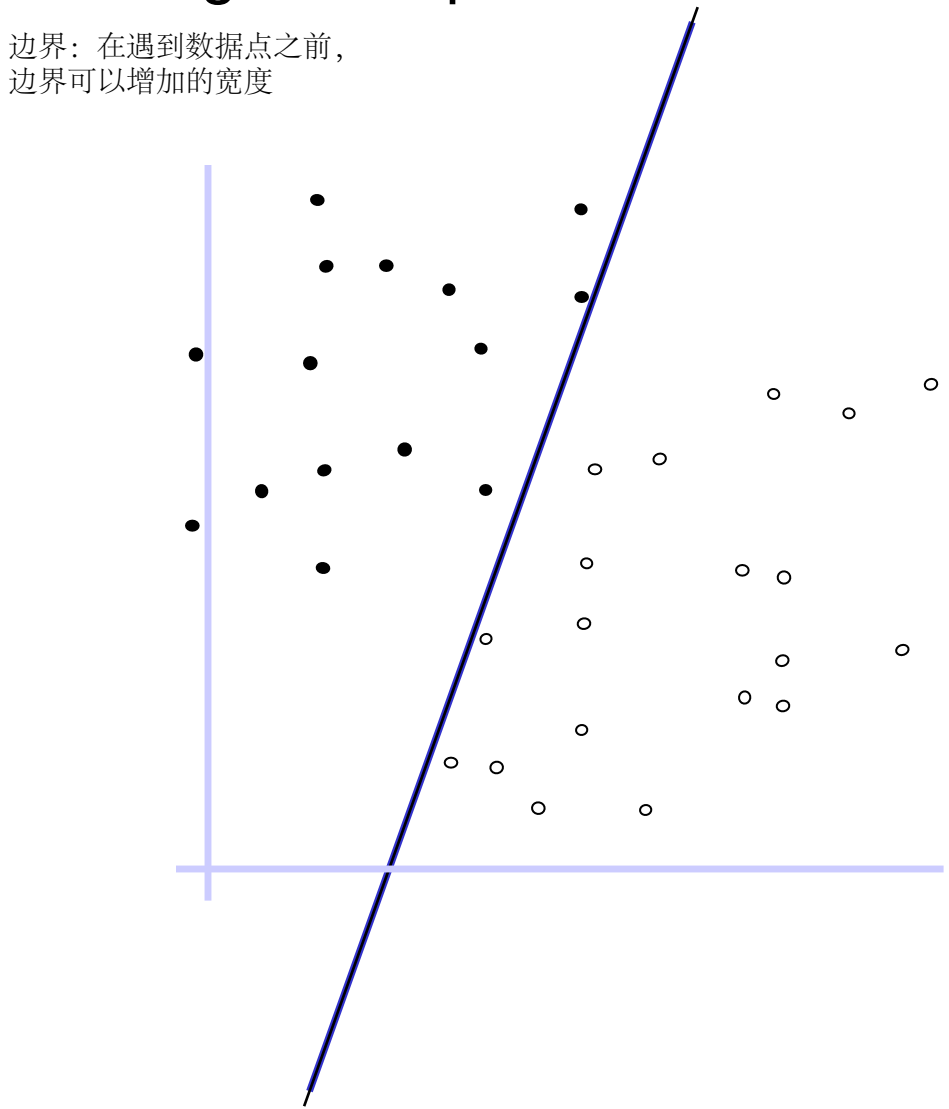


Any of these would be fine..

..but which is the best?

The margin

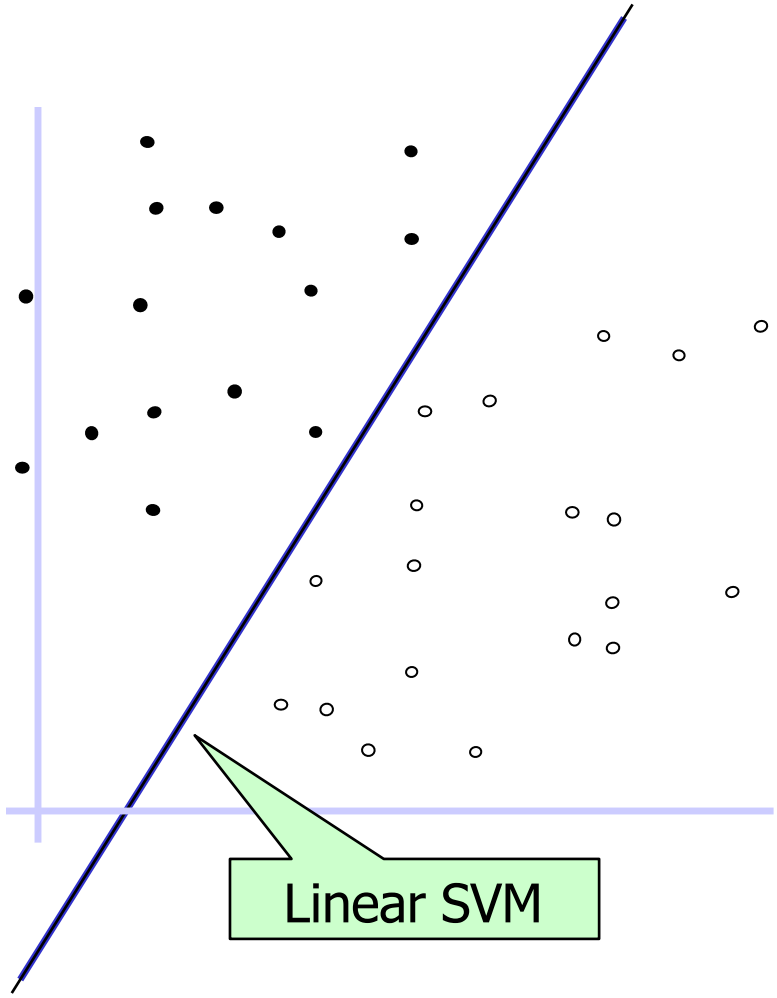
- Margin: the width that the boundary can be increased before hitting a data point



SVM: maximize margin

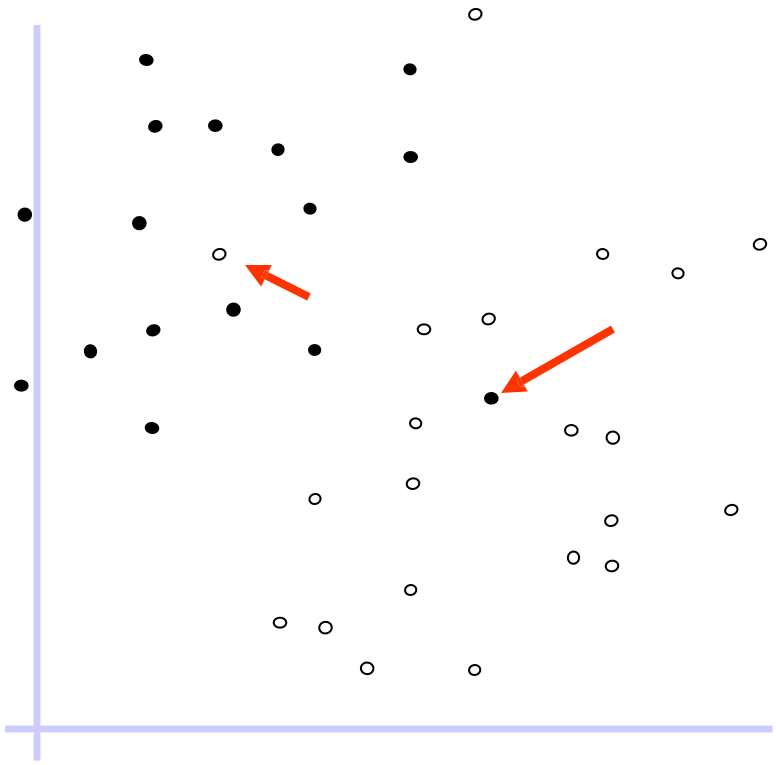
- The simplest SVM (linear SVM) is the linear classifier with the maximum margin

最简单的SVM（线性SVM）是具有最大边际的线性分类器



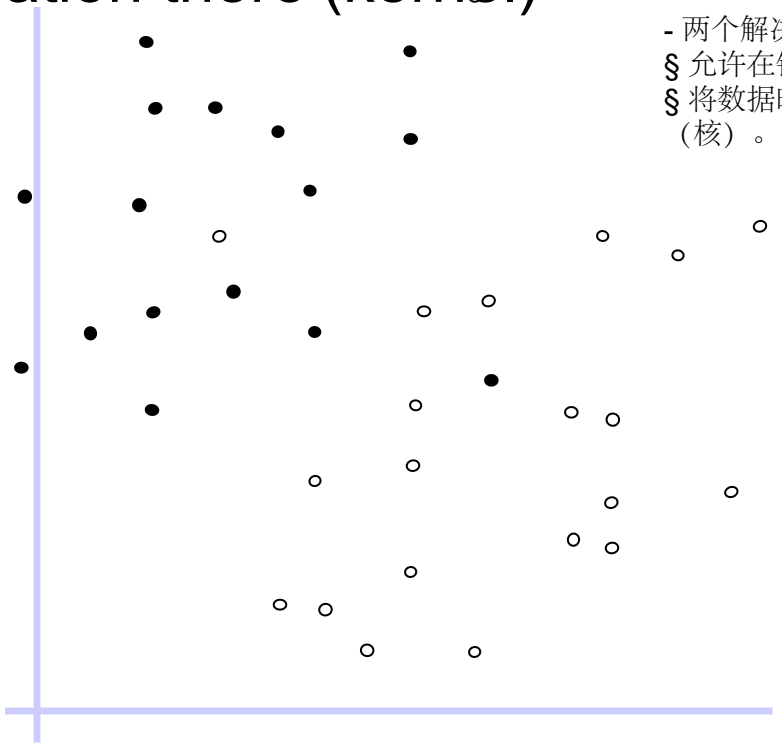
SVM: linearly non-separable data

- What if the data is not linearly separable?



SVM: linearly non-separable data

- Two solutions:
  - Allow a few points on the wrong side (slack variables), and/or
  - Map data to a higher dimensional space, do linear classification there (kernel)



- 两个解决方案。
§ 允许在错误的一侧有几个点（松弛变量），和/或
§ 将数据映射到一个高维空间，在那里做线性分类（核）。

SVM: more than two classes

- N class problem: Split the task into N binary tasks:
  - Class 1 vs. the rest (class 2—N)
  - Class 2 vs. the rest (class 1, 3—N)
  - …
  - Class N vs. the rest
- Finally, pick the class that put the point furthest into the positive region.

- N类问题：将任务分成N个二元任务。
§ 第1类与其他类（第2-N类）
§ 第2类对其余的（第1类，3-N）。
§ ...
§ 第N类与其余的对比
- 最后，挑选出把点放到正区域最远的那一类。

SVM: get your hands on it

- There are many implementations
- http://www.support-vector.net/software.html
- http://svmlight.joachims.org/

- You don't have to know the rest of the class in order to use SVM.

end of class

(But you want to know more, don't you?)

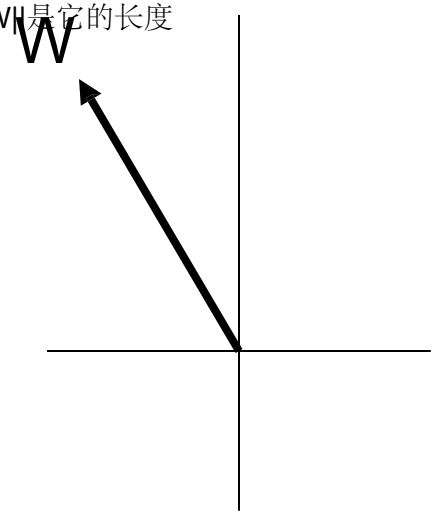
The math version

Support vector machines

Vector

- A vector W in *d*-dimensional space is a list of *d* numbers, e.g. W=(-1,2)’
- A vector is a vertical column. ’ is matrix transpose
- Vector is a line segment, with arrow, in space
- The norm of a vector ||W|| is its length

- 在d维空间的一个向量W是一个d个数字的列表，例如，W=(-1,2)'
- 一个向量是一个纵列。'是矩阵转置
- 向量是空间中的一条线段，带有箭头。
- 一个向量的规范||W||是它的长度



$$||W|| = \text{sqrt}(W'W)$$

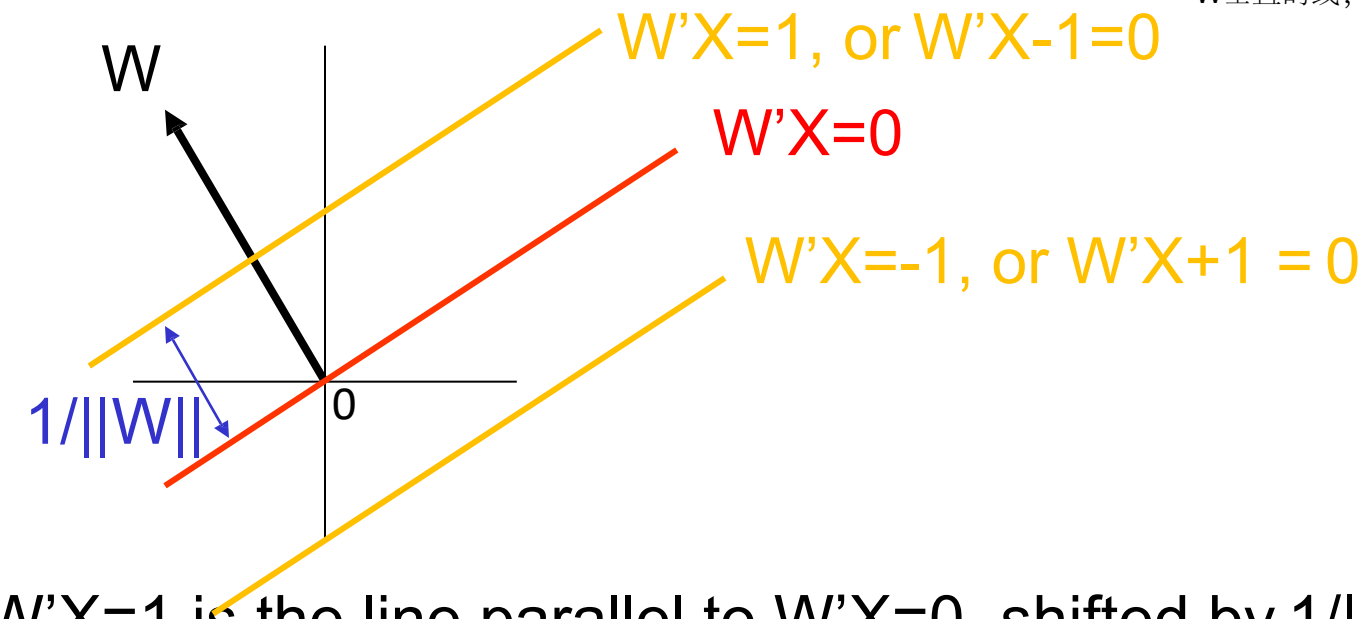
Inner product: $X'Y = \sum X_i Y_i$

What d-dimensional point X makes W’ X = 0?

Lines

- W' X is a scalar (single number): X's projection onto W
- W'X=0 specifies the set of points X, which is the line perpendicular to W, and intersects at (0,0)

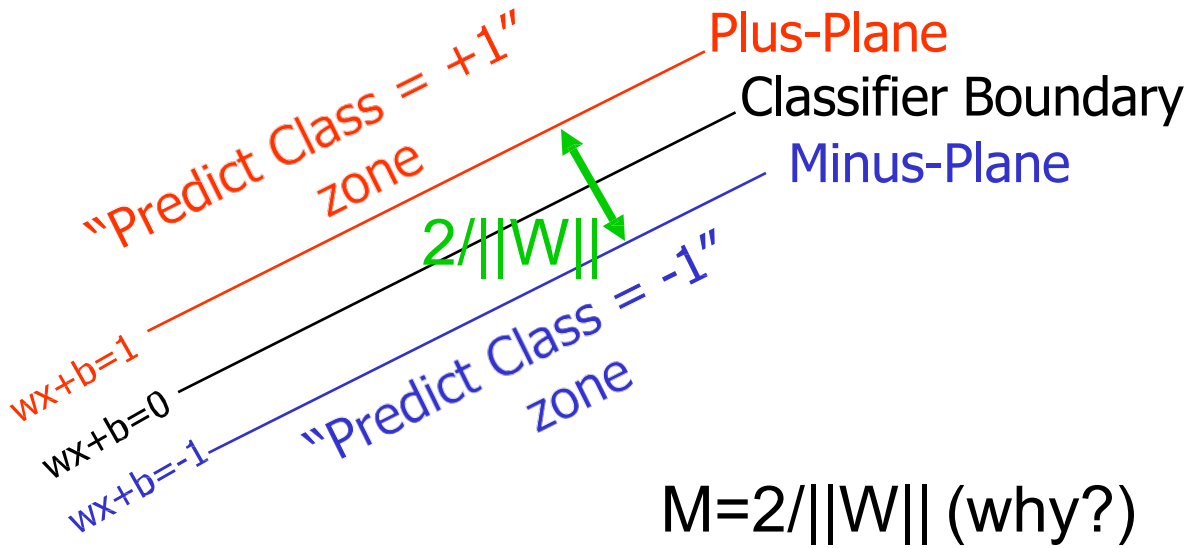
- W' X是一个标量（单数）。X对W的投影
- W'X=0指定点X的集合，它是与W垂直的线，并相交于(0,0)。



- W'X=1 is the line parallel to W'X=0, shifted by 1/||W||
- What if the boundary doesn't go through origin?

SVM boundary and margin

- Want: find W, b (offset) such that
  - all positive training points (X,Y=1) in the red zone,
  - all negative ones (X, Y=-1) in the blue zone,
  - the margin M is maximized

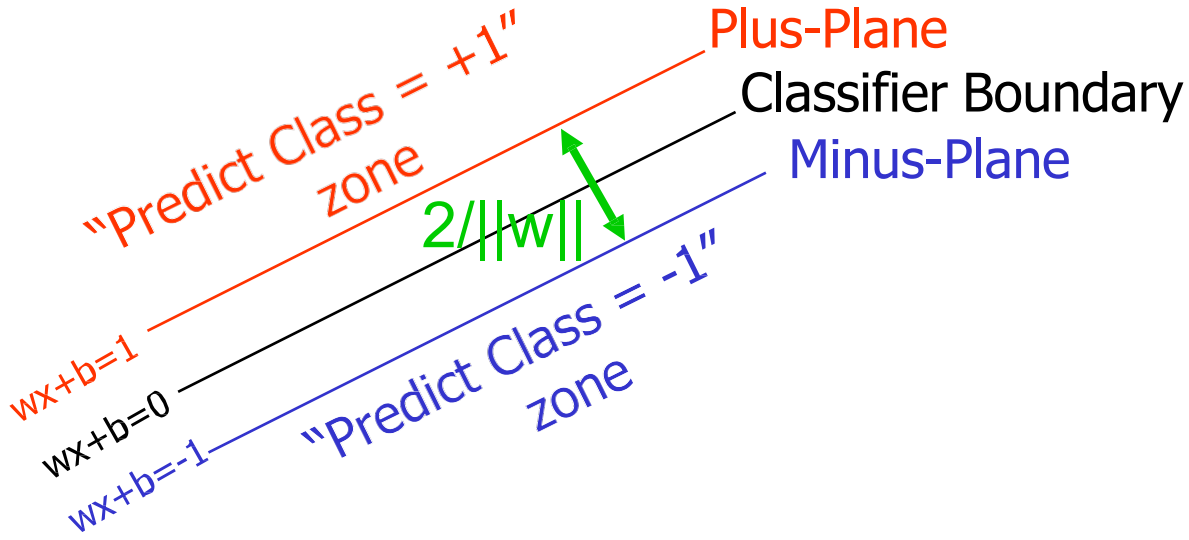


M=2/||W|| (why?)
How do we find such W,b?

SVM as constrained optimization

- Variables: W, b
- Objective function: maximize the margin M=2/||W||
- Equiv. to minimize ||W||, or $||W||^2$=W'W, or ½W'W

- Assume N training points (Xi, Yi), Yi = 1 or -1
- Subject to each training point on the correct side (the constraint). How many constraints do we have?



SVM as constrained optimization

- Variables: W, b
- Objective function: maximize the margin M=2/||W||
- Equiv. to minimize ||W||, or $||W||^2$=W'W, or ½W'W

- 变量。W, b
- 目标函数：最大化保证金M=2/||W||||
- 等于最小化||W||，或||W||2=W'W，或 ½W'W

- Assume N data points (Xi, Yi), Yi = 1 or -1
- Subject to each training point on the correct side (the constraint). How many constraints do we have? N
  - W'Xi + b >= 1, if Yi=1
  - W'Xi + b <= -1, if Yi=-1
  - we can unify them: Yi(W'Xi+b) >= 1

我们有一个连续约束的优化问题！我们要做什么？我们该怎么做？

- We have a continuous constrained optimization problem! What do we do?

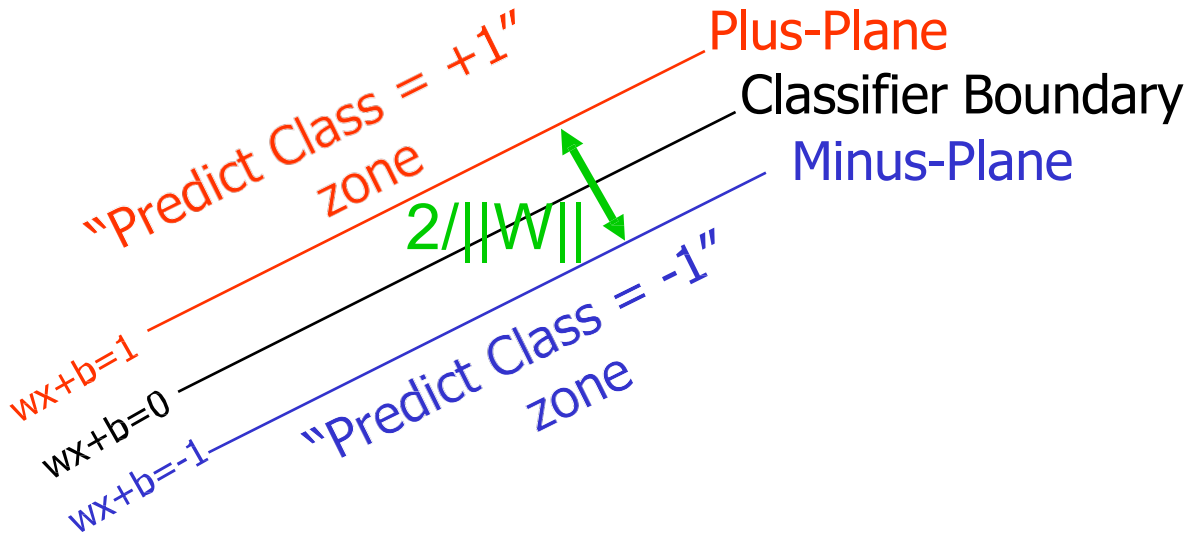
SVM as QP

$$\min_{W,b} \tfrac{1}{2} W'W$$

Subject to $Y_i (W'X_i + b) >= 1$, for all i

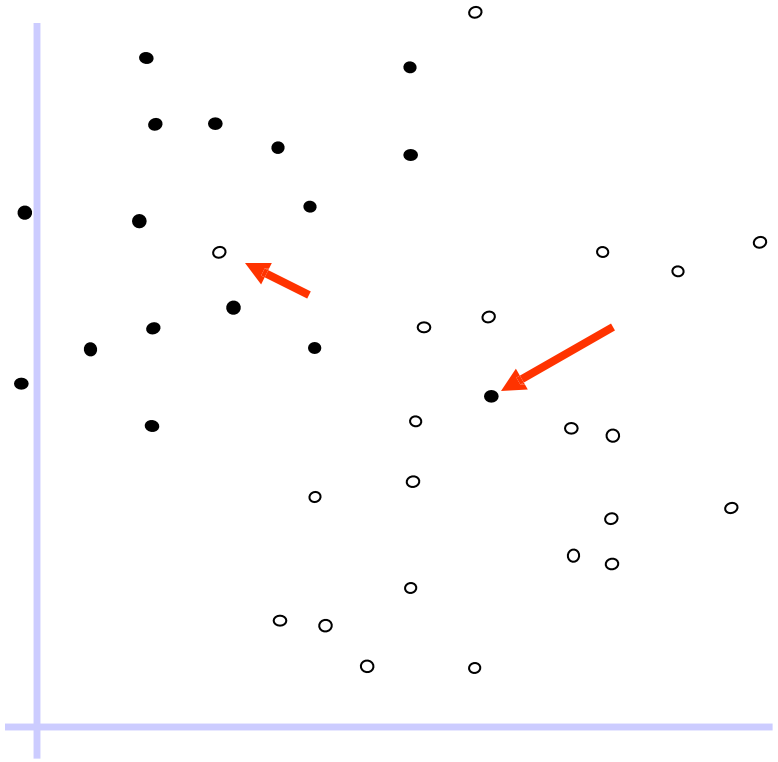
- Objective is convex, quadratic
- Linear constraints
- This problem is known as Quadratic Program (QP), for which efficient global solution algorithms exist.

- 目标是凸的，是二次的
- 线性约束
- 这个问题被称为二次方程序（QP），存在有效的全局解决算法。



Non-separable case

- What about this?

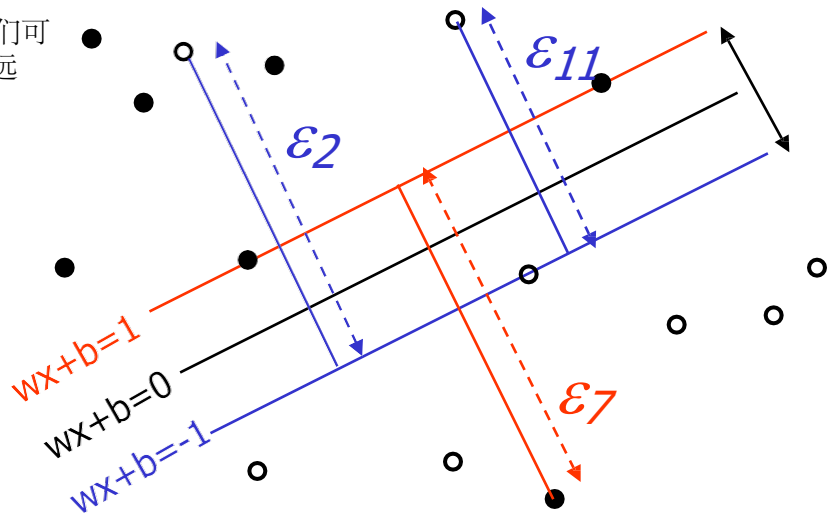


Can we insist on Yi (W'Xi + b) >= 1, for all i?

Trick #1: slack variables

- Relax the constraints – allow a few “bad apples”
- For a given linear boundary W, b, we can compute how far off into the wrong side a bad point is

- 放宽约束--允许一些 "坏苹果"
- 对于一个给定的线性边界W，b，我们可以计算出一个坏点在错误的一侧有多远



- Here’s how we relax the constraints:

$$Yi\ (W'Xi + b) >= 1 - \varepsilon_i$$

Trick #1: SVM with slack variables

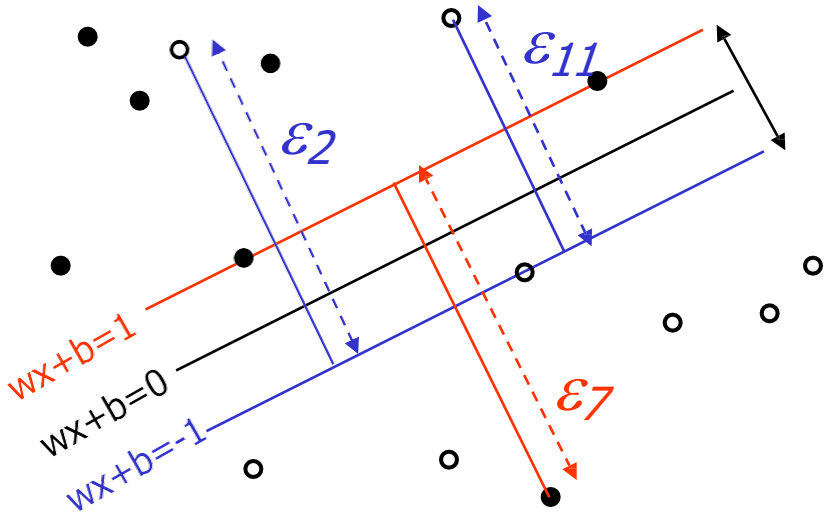
$$\min_{W,b,\varepsilon} \tfrac{1}{2}\, W'W + C \sum_i \varepsilon_i$$

Trade-off parameter

Subject to

$$Y_i\,(W'X_i + b) >= 1 - \varepsilon_i \text{ for all } i$$

$$\varepsilon_i >= 0 \text{ for all } i \text{ (why?)}$$



Trick #1: SVM with slack variables

$$\min_{W,b,\varepsilon} \tfrac{1}{2}\, W'W + C \sum_i \varepsilon_i$$

Trade-off parameter

Subject to

$$Y_i\,(W'X_i + b) >= 1 - \varepsilon_i \text{ for all } i$$

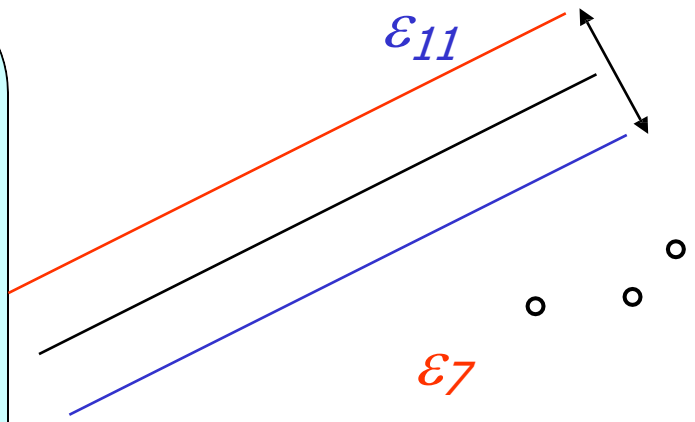
$$\varepsilon_i >= 0 \text{ for all } i$$

Originally we optimize variables W (d-dimensional vector), b

Now we optimize W, b, $\varepsilon_1 \ldots \varepsilon_N$

Now we have 2N constraints

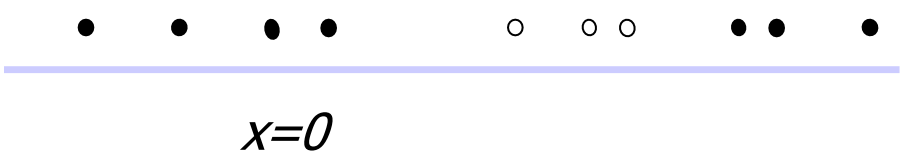
Still a QP (**soft-margin SVM**)



Another look at non-separable case

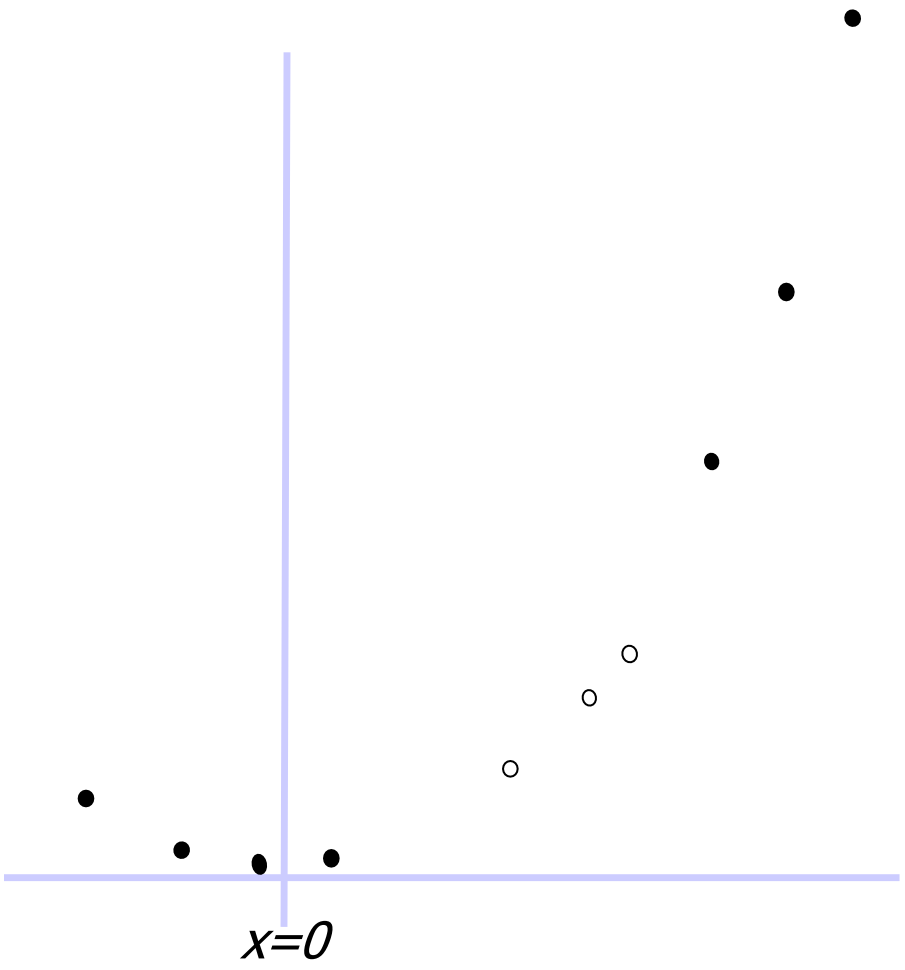
- Here's another non-separable dataset, in 1-dimensional space
- We can of course use slack variables… but here's another trick!

- 这是另一个不可分割的数据集，在一维空间中
- 我们当然可以使用松弛变量.....但这里有另一个技巧!



Trick #2: Map data to high dimensional space

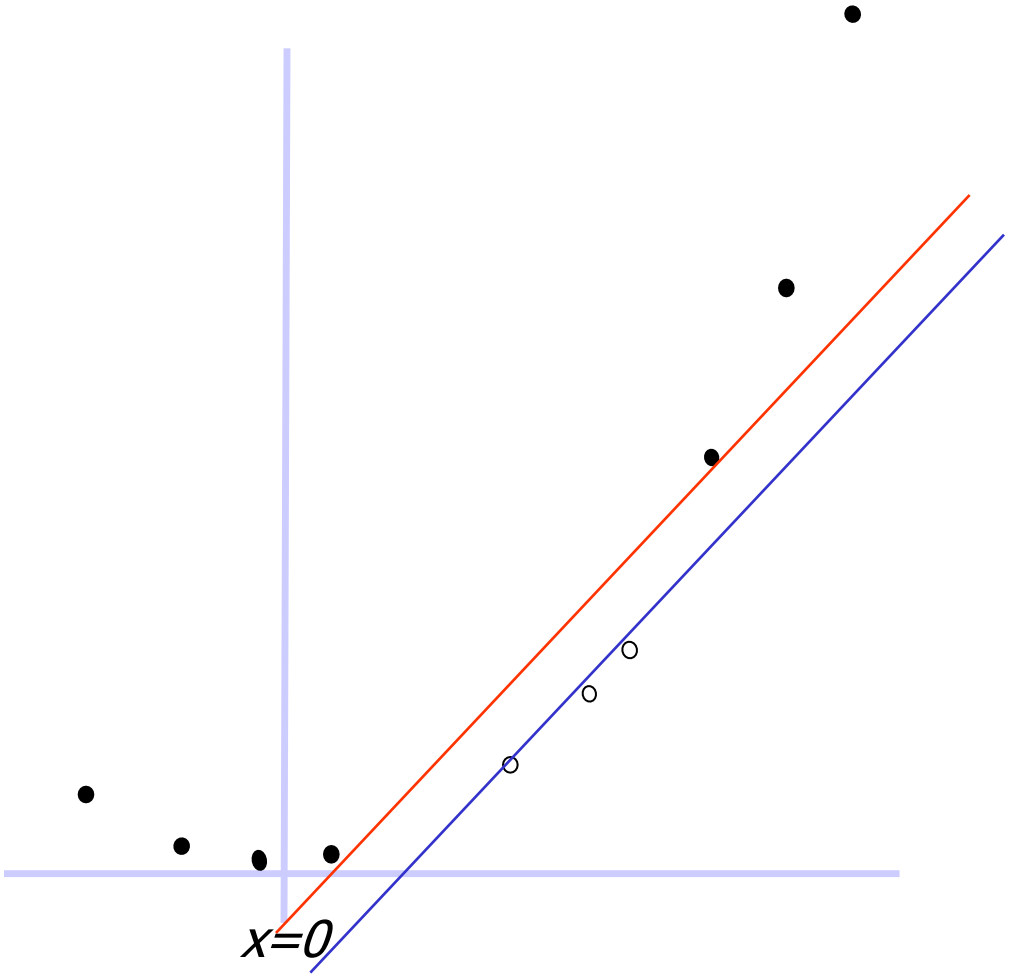
- We can map data x from 1-D to 2-D by $x \rightarrow (x, x^2)$



Trick #2: Map data to high dimensional space

- We can map data x from 1-D to 2-D by

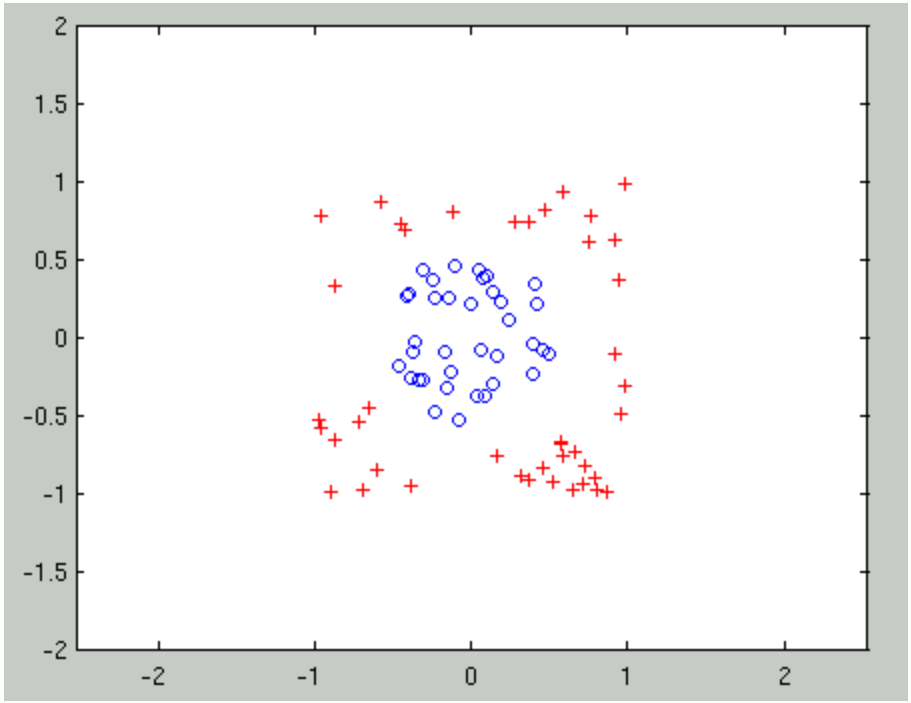
$$x \rightarrow \Phi(x)=(x, x^2)$$

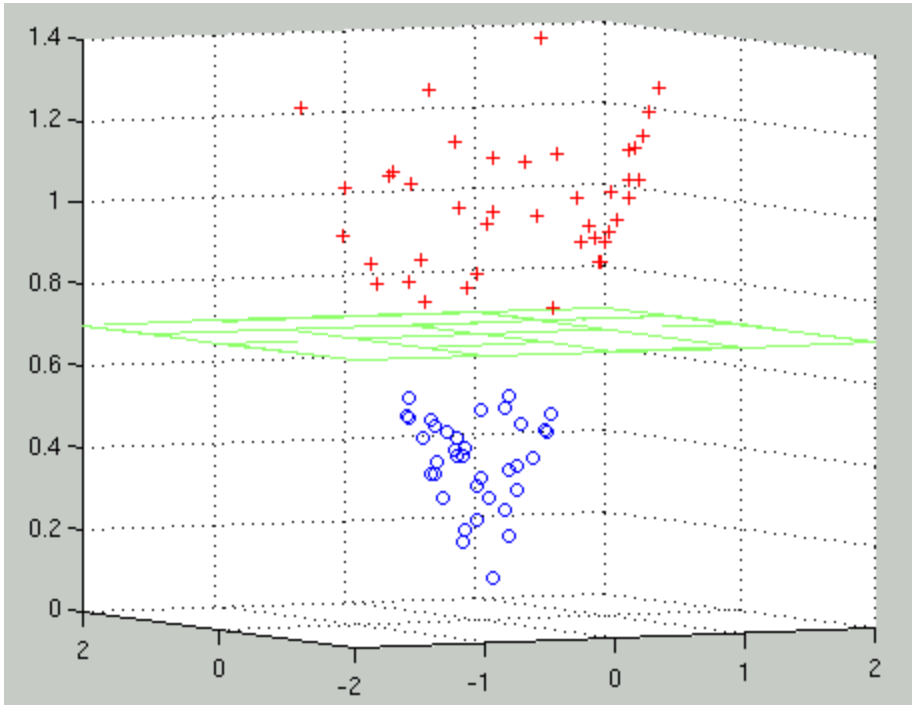


•Now the data is linearly separable in the new space!
•We can run SVM in the new space
•The linear boundary in the new space corresponds to a non-linear boundary in the old space

Another example

$$(x_1, x_2) \Rightarrow (x_1, x_2, \sqrt{x_1^2 + x_2^2})$$

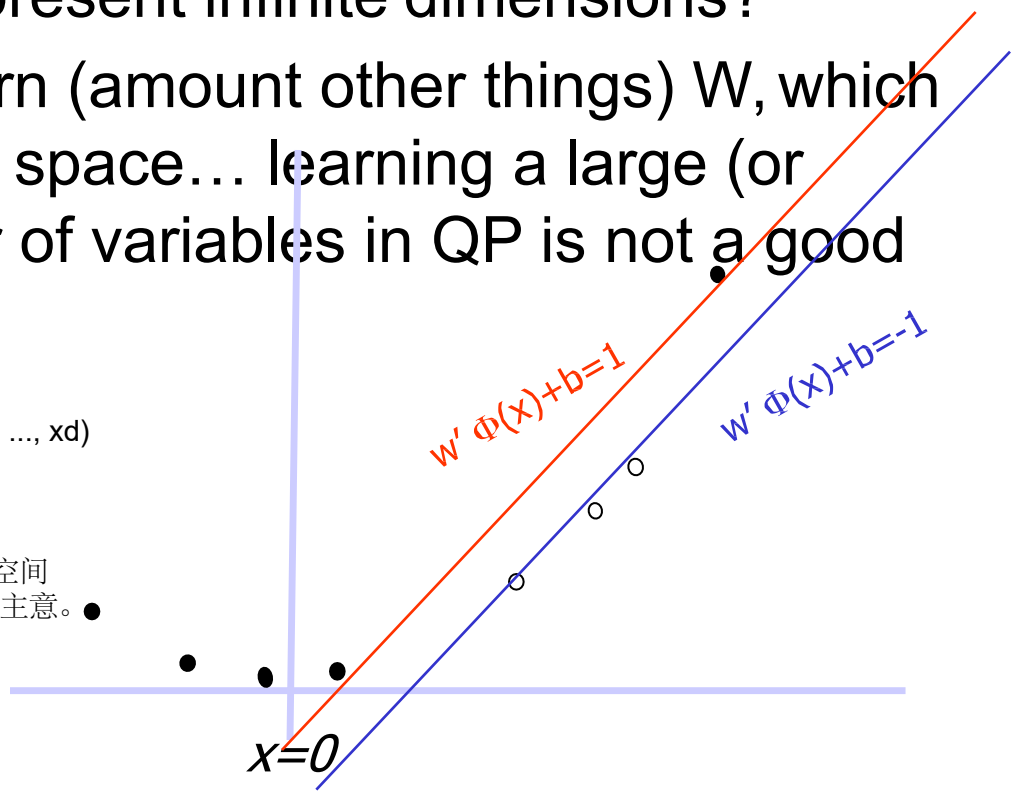




Trick #2:Map data to high dimensional space

- In general we might want to map an already high-dimensional $X=(x_1, x_2, \ldots, x_d)$ into some much higher, even infinite dimensional space $\Phi(x)$
- Problems:
  - How do you represent infinite dimensions?
  - We need to learn (amount other things) W, which lives in the new space… learning a large (or infinite) number of variables in QP is not a good idea.

- 一般来说，我们可能想把一个已经是高维的X=(x1, x2, ..., xd)映射到一些更高的，甚至是无限维的空间F(x)
- 问题是。
§ 如何表示无限的维度？
§ 我们需要学习（相当于其他东西）W，它生活在新的空间中.....在QP中学习大量（或无限）的变量并不是一个好主意。



Trick #2: kernels

- We will do several things:
  - Convert it into a equivalent QP problem, which does not use W, or even $\Phi(X)$ alone!
    - It only uses the inner product $\Phi(X)'\ \Phi(Y)$, where X and Y are two training points. The solution also only uses such inner product
    - It still seems infeasible to compute for high (infinite) dimensions
  - But there are smart ways to compute such inner product, known as kernel (a function on two var.)
    - kernel(X,Y) $\Leftrightarrow$ inner product $\Phi(X)'\ \Phi(Y)$
- Why should you care:
  - One kernel, one new (higher dimensional) space
  - You will impress friends at cocktail parties

- 我们将做几件事。
§ 将其转换成一个等价的QP问题，这个问题不使用W，甚至不单独使用F(X)!
它只使用内积F(X)' F(Y)，其中X和Y是两个训练点。该解决方案也只使用这样的内积
- 对于高（无限）维度来说，它的计算似乎仍然不可行
§ 但有一些聪明的方法来计算这样的内积，称为内核（两个变量上的一个函数）。
- kernel(X,Y) óinner product F(X)' F(Y)
你为什么要关心
§ 一个内核，一个新的（高维）空间
§ 你会在鸡尾酒会上给朋友留下深刻印象

Prepare to bite the bullet…

- Here's the original QP formula

$$\min_{W,b} \tfrac{1}{2}\, W'W$$

Subject to $Y_i (W'X_i + b) >= 1$, for all i (N constraints)

- Remember Lagrange multipliers?

$$L = \tfrac{1}{2}\, W'W - \sum a_i [Y_i (W'X_i + b) - 1]$$

Subject to $a_i >= 0$, for all i

We have them here because those are inequality constraints

- We want the gradient of L to vanish with respect to W, b, a. Try this and you'll get

$$W = \sum a_i Y_i X_i$$

$$\sum a_i Y_i = 0$$

Put them back into the Lagrangian L

Biting the bullet

$$\max_{\{ai\}} \sum ai - \tfrac{1}{2} \sum_{i,j} ai\ aj\ Yi\ Yj\ \mathbf{Xi'\ Xj}$$

Subject to

$$ai >= 0, \text{ for all I}$$

$$\sum ai\ Yi = 0$$

- This is an equivalent QP problem (the dual)
- Before we optimize W (d variables), now we optimize a (N variables): which is better?
- X only appears in the inner product

- 这是一个等价的QP问题（对偶）。
- 之前我们优化W（d变量），现在我们优化a（N变量）：哪个更好？
- X只出现在内积中

Biting the bullet

$$\max_{\{ai\}} \sum ai - \tfrac{1}{2} \sum_{i,j} ai\ aj\ Yi\ Yj\ \Phi(Xi)' \Phi(Xj)$$

Subject to

$$ai >= 0, \text{ for all I}$$

$$\sum ai\ Yi = 0$$

If we map X to new space $\Phi(X)$

- This is an equivalent QP problem
- Before we optimize W (d variables), now we optimize a (N variables): which is better?
- X only appears in the inner product

What's special about kernel

- Say data is two dimensional: s=(s1,s2)
- We decide to use a particular mapping into 6 dimensional space

$$\Phi(s)=(s1^2, s2^2, \sqrt{2}\, s1s2, s1, s2, 1)$$

- Let another point be t=(t1,t2).

$$\Phi(s)' \Phi(t)=s1^2\, t1^2 + s2^2\, t2^2 + 2\, s1s2t1t2 + s1t1 + s2t2 + 1$$

– 假设数据是二维的：$\Sigma=(\sigma1,\sigma2)$
– 我们决定使用一个特定的六维空间的映射

What's special about kernel

- Say data is two dimensional: s=(s1,s2)
- We decide to use a particular mapping into 6 dimensional space

$$\Phi(s)=(s1^2, s2^2, \sqrt{2}\, s1s2, \sqrt{2}\, s1, \sqrt{2}\, s2, 1)$$

- Let another point be t=(t1,t2).

$$\Phi(s)' \Phi(t)=s1^2\, t1^2 + s2^2\, t2^2 + 2\, s1s2t1t2 + 2\, s1t1 + 2\, s2t2 + 1$$

- Let the kernel be $K(s,t) = (s't+1)^2$
- Verify that they are the same. We saved computation.

Kernels

- You ask: “Is there such a good K for any Φ I pick?”
- The inverse question: “Given some K, is there a Φ so that K(X,Y) = Φ(X)’ Φ(Y)?”
- Mercer’s condition: the inverse question is true…

$$\text{if for any } g(\mathbf{x}) \text{ such that } \int g(\mathbf{x})^2 d\mathbf{x} \text{ is finite}$$

$$\int K(\mathbf{x},\mathbf{y}) g(\mathbf{x}) g(\mathbf{y}) d\mathbf{x} d\mathbf{y} \geq 0.$$

- This is positive semi-definiteness, if you must know
- Φ may be infinite dimensional; we may not be able to explicitly write down Φ

- 这就是正半定义性，如果你一定要知道的话
- F可能是无限维的；我们可能无法明确地写下F

Some frequently used kernels

- Linear kernel: $K(X,Y) = X'Y$
- Quadratic kernel: $K(X,Y) = (X'Y+1)^2$
- Polynomial kernel: $K(X,Y) = (X'Y+1)^n$
- Radial Basis Function kernel: $K(X,Y) = \exp(-||X-Y||^2/\sigma)$
- Many, many other kernels
- Hacking with SVM: create various kernels, hope their space Φ is meaningful, plug them into SVM, pick the one with good classification accuracy (equivalent to feature engineering)
- Kernel summary: QP of size N, nonlinear SVM in the original space, new space possibly high/infinite dim, efficient if K is easy to compute
- Kernel can be combined with slack variables

- 线性内核。K(X,Y) = X'Y
- 二次方内核。K（X，Y）=（X'Y+1）2
- 多项式内核。K(X,Y) = (X'Y+1)n
- 径向基函数核。K(X,Y) = exp(- ||X-Y||2/s)
- 很多很多其他内核
- 用SVM进行黑客攻击：创建各种核，希望它们的空间F是有意义的，把它们插入SVM，挑选出分类精度高的核（相当于特征工程）。
- 内核总结。大小为N的QP，在原空间中的非线性SVM，新空间可能是高/无限的dim，如果K容易计算，则效率很高
- 内核可以与松弛变量相结合

Why the name "support vector machines"

$$\max_{\{ai\}} \sum ai - \tfrac{1}{2} \sum_{i,j} ai\ aj\ Yi\ Yj\ \mathbf{K(Xi, Xj)}$$

Subject to

$$ai >= 0, \text{ for all I}$$

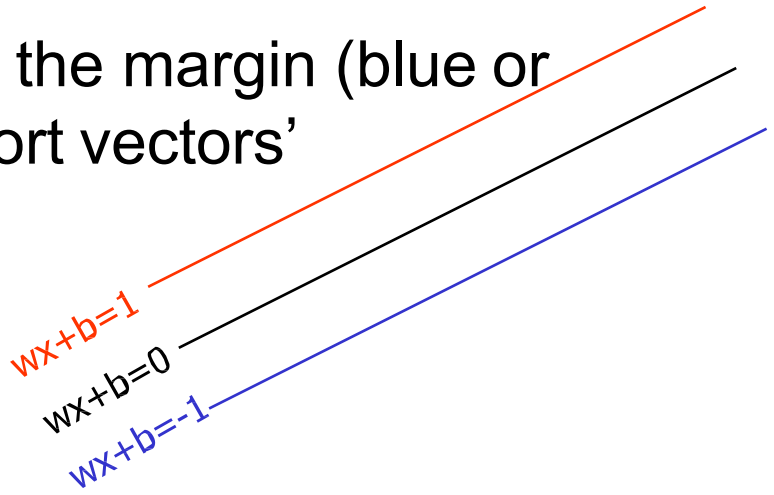
$$\sum ai\ Yi = 0$$

- The decision boundary is

$$f(Xnew)=W' Xnew + b= \sum ai\ Yi\ Xi' Xnew + b$$

- In practice, many *a*'s will be zero in the solution!
  - Those few X with a>0 lies on the margin (blue or red lines), they are the 'support vectors'

- 在实践中，许多a会在解中为零!
§ 那几个a>0的X位于边缘（蓝线或红线）上，它们是 "支持向量



What you should know

- The intuition, where to find software
- Vector, line, length
- Margin
- QP with linear constraints
- How to handle non-separable data
  - Slack variables
  - Kernels ⇔ new feature space

- Ref: **A Tutorial on Support Vector Machines for Pattern Recognition (1998)** Christopher J. C. Burges